

第 5 章 CSS 布局

网页本质上是一种在浏览器上完成的平面设计作品,网页布局就是将网页元素合理排列在网页版面上,达到美观大方、井然有序的效果。使用 CSS 进行网页布局,本质是利用标准流、浮动或定位属性的性质对网页布局元素进行合理排列。

5.1 浮动

在标准流中,块级元素的盒子都是上下排列,行内元素的盒子都是左右排列,如果仅仅按照标准流的方式进行排列,就只有这几种可能性,限制太大。CSS 的制定者也想到了这样排列限制的问题,因此又给出了浮动和定位方式,从而使排版的灵活性大大提高。

如果希望相邻的块级元素盒子左右排列(所有盒子浮动)或者希望一个盒子被另一个盒子中的内容所环绕(一个盒子浮动)做出图文混排的效果,这时最简单的实现办法就是运用浮动(float)属性使盒子在浮动方式下定位。

5.1.1 盒子浮动后的特点

在标准流中,一个块级元素在水平方向会自动伸展,在它的父元素中占满整个一行;而在竖直方向和其他元素依次排列,不能并排,如图 5-1 所示。使用“浮动”方式后,这种排列方式就会发生改变。



图 5-1 3 个盒子在标准流中

CSS 中有一个 float 属性,默认值为 none,也就是标准流通常的情况,如果将 float 属性的值设为 left 或 right,元素就会向其父元素的左侧或右侧靠近,同时盒子的宽度不再伸展,而是收缩,在没设置宽度时,会根据盒子里面的内容来确定宽度。

下面通过一个实验来演示浮动的作用,基础代码(5-1.html)如下,这个代码中没有使用浮动,它的显示效果如图 5-1 所示。

```
div{
    padding:10px; margin:10px; border:1px dashed #111;
    background-color:#90baff;    }
.father{
    background-color:#ff9;  border:1px solid #111;  }
<div class="father">
    <div class="son1">Box-1</div>
    <div class="son2">Box-2</div>
    <div class="son3">Box-3</div>
</div>
```

1. 一个盒子浮动

接下来在上述代码中添加一条 CSS 代码,使 Box-1 盒子浮动。代码(5-2. html)如下:

```
.son1{float:left;}
```

此时显示效果如图 5-2 所示,可发现在给 Box-1 添加浮动属性后,Box-1 的宽度不再自动伸展,而且不再占据原来浏览器分配给它的位置。如果再在未浮动的盒子 Box-2 中添一行文本,就会发现 Box-2 中的内容是环绕着浮动盒子的,如图 5-3 所示。



图 5-2 第一个盒子浮动



图 5-3 增加第二个盒子的内容

总结: 设置元素浮动后,元素发生了如下一些改变:

- (1) 浮动后的盒子将以行内块(inline-block)元素显示,即宽度会自动收缩,但保持块级元素的其他性质。
- (2) 浮动的盒子将脱离标准流,即不再占据浏览器原来分配给它的位置。
- (3) 未浮动的盒子将占据浮动盒子的位置,同时未浮动盒子内的内容会环绕浮动后的盒子。

提示: 所谓“脱离标准流”,是指元素不再占据在标准流下浏览器分配给它的空间,其他元素就好像这个元素不存在一样。例如在图 5-2 中,当 Box-1 浮动后,Box-2 就顶到了 Box-1 的位置,相当于 Box-2 视 Box-1 不存在一样。但是,浮动元素并没有完全脱离标准流,这表现在浮动盒子会影响未浮动盒子中内容的排列,例如 Box-2 中的内容会跟在 Box-1 盒子之后进行排列,而不会忽略 Box-1 盒子的存在。

2. 多个盒子浮动

在 Box-1 浮动的基础上再设置 Box-2 也左浮动,代码(5-3.html)如下:

```
.son2{float:left;}
```

此时显示效果如图 5-4 所示(在 Box-3 中添加了一行文本)。可发现 Box-2 盒子浮动后仍然遵循上面浮动的规律,即 Box-2 的宽度也不再自动伸展,而且不再占据原来浏览器分配给它的位置。

如果将 Box-1 的浮动方式改为右浮动: `.son1{float:right}`,则显示效果如图 5-5 所示,可看到 Box-2 移动到了 Box-1 的前面,这说明元素浮动后其显示顺序和它们在代码中的位置可能并不一致。

接下来设置 Box-3 也左浮动: `.son3{float:left}`,则显示效果如图 5-6 所示。可发现 3 个盒子都浮动后,就产生了块级元素水平排列的效果。同时由于都脱离了标准流,导致其父元素中的内容为空。



图 5-4 设置两个盒子浮动



图 5-5 改变浮动方向



图 5-6 3 个盒子都浮动

对于多个盒子浮动,除了每个浮动盒子都遵循单个盒子浮动的规律外,还有以下两条规律:

(1) 多个浮动元素不会相互重叠,一个浮动元素的外边界(margin)碰到另一个浮动元素的外边界后便停止运动。

(2) 若包含的容器太窄,无法容纳水平排列的多个浮动元素,那么最后的浮动盒子会向下移动(见图 5-7)。但如果浮动元素的高度不同,那当它们向下移动时可能会被卡住(见图 5-8)。

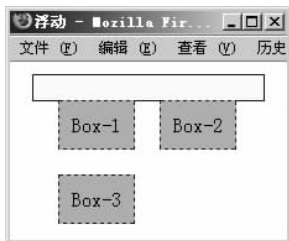


图 5-7 没有足够的水平空间

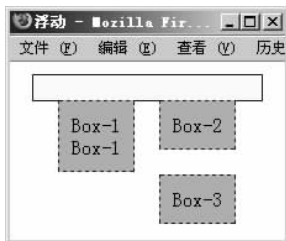


图 5-8 被 Box-1 卡住了

5.1.2 清除浮动元素的影响

clear 是清除浮动属性,它的取值有 left、right、both 和 none(默认值)。如果设置盒子的 clear 属性值为 left 或 right,表示该盒子的左边或右边不允许有浮动的元素。值设置为 both 则表示两边都不允许有浮动元素,因此该盒子将会在浏览器中另起一行显示。

例如,在图 5-5 两个盒子浮动的基础上,设置 Box-3 清除浮动,即在 5-1.html 基础上设置如下 CSS 代码(5-4.html),效果如图 5-9 所示。

```
.son1{float:right;}      .son2{float:left;}
.son3{clear:both;}
```

可以看到,对 Box-3 清除浮动(clear:both;),表示 Box-3 的左右两边都不允许有浮动的元素,因此 Box-3 移动到了下一行显示。

实际上,clear 属性既可以用在未浮动的元素上,也可以用在浮动的元素上,如果对 Box-3 同时设置清除浮动和浮动,即:

```
.son3{clear:both; float:left;}
```

则效果如图 5-10 所示,可看到 Box-3 的左右仍然没有浮动的元素。

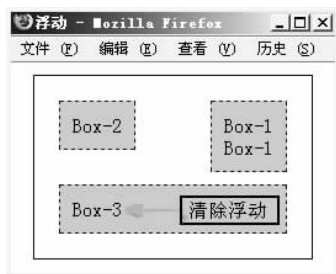


图 5-9 对 Box-3 清除浮动

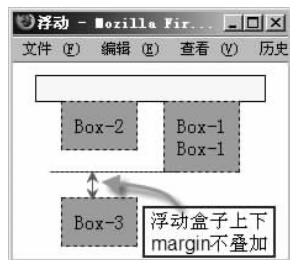


图 5-10 对 Box-3 设置清除浮动和浮动

由此可见,清除浮动是清除其他盒子浮动对该元素的影响,而设置浮动是让元素自身浮动,两者并不矛盾,因此可同时设置元素清除浮动和浮动。

由于上下 margin 叠加现象只会发生在标准流的情况下,而浮动方式下盒子的任何 margin 都不会叠加,所以设置盒子浮动并清除浮动后,上下两个盒子的 margin 不叠加。在图 5-10 中,Box-3 到 Box-1 之间的垂直距离是 20px,即它们的 margin 之和。

5.1.3 清除浮动影响的方法

在网页布局中,浮动元素会影响其他元素的正常排列,导致网页元素不能按人们的意愿显示。因此设置元素浮动后应尽量清除该元素浮动对其他元素的影响。对于下列两种

情形应采用不同的方法清除浮动。

1. 来自子元素的浮动

如果一个父元素内所有的子元素都浮动,一定要记得对这个父元素作清除浮动处理。否则该父元素下面的元素会顶到浮动元素的位置上去(如图 5-11 所示),下面的代码(5-5. html)可清除子元素浮动对父元素的影响,其显示效果如图 5-12 所示。

```
div{padding:10px; margin:10px;
    border:1px dashed #111;    background-color:#9bf; }
.father{background-color:#ff9; border:1px solid #111; }
.cls {clear: both; }
.son1{float:right; }        .son2{float:left;}
.box3{background:#ccf;}

<div class="father">
    <div class="son1">Box-1<br />Box-1</div>
    <div class="son2">Box-2</div>
    <br class="cls">                                <!--这是清除浮动的元素-->
</div>
<div class="box3">Box-3</div>                        <!--父元素下面的元素-->
```

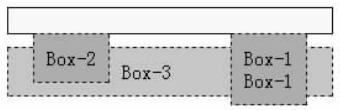


图 5-11 未清除子元素浮动时的效果

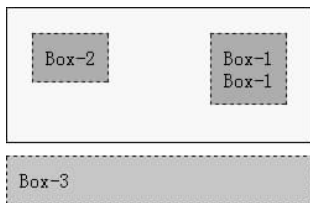


图 5-12 清除子元素浮动后的效果

这种方法添加了一个元素(.cls)专门用来清除浮动,如果不愿添加新元素,可使用伪元素的方式来设置清除浮动。下面的代码在父元素内部的末尾添加一个伪元素,使用伪元素清除浮动的经典代码(5-6. html)如下:

```
.father:after { content: ''; display: table; clear: both; }
.father { * zoom: 1; }                                /* 兼容 IE6、IE7,如不需要可去掉 */
```

注意: 对于一个父元素内的所有子元素都浮动,一种极其错误的做法是设置父元素的高度来掩饰浮动对它的影响,如 .father{height:50px;} ,这样做只是掩饰了浮动,并没有清除浮动的影响,虽然使父元素看起来正常了,但父元素下面的元素仍然会顶到上面去。

因此,在 CSS 布局时,如果发现一个元素移动到它原来位置的左上方或右上方,并且和其他元素发生了重叠,则很可能是受到了其他盒子浮动的影响。

2. 来自兄弟元素的浮动

如果一个元素前面的兄弟元素浮动(见图 5-13),则可以对紧邻该浮动元素的后一个元素作清除浮动处理。例如,不希望 Box-3 受前面兄弟元素浮动的影响,则可对 Box-3 清除浮动,代码(5-7. html)如下,显示效果如图 5-14 所示。

```
div{
    padding:10px 20px; margin:7px; border:1px dashed #111;
    background-color:#9bf; float:left; } /* 所有 div 都浮动 */
.box3{background:#ccf; clear:both;} /* 对 Box-3 清除浮动 */
<div class="son1">Box-1<br />Box-1</div>
<div class="son2">Box-2</div>
<div class="box3">Box-3</div><div class="box4">Box-4</div>
```

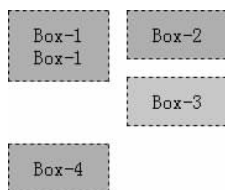


图 5-13 清除浮动前

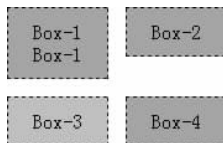


图 5-14 清除浮动后

除了直接对后面的元素清除浮动外,还可以增加一个元素专用于清除浮动,代码如下:

```
<div class="son1">Box-1<br />Box-1</div><div class="son2">Box-2</div>
<br class="cls" style="clear:both" > <!-- 增加一个清除浮动的元素 -->
<div class="box3">Box-3</div><div class="box4">Box-4</div>
```

虽然增加一个元素使代码变得冗余,但这样使每个元素的功能变得更加清晰,因此推荐使用。

3. 避免滥用浮动

由于元素浮动会对其他元素的布局产生影响,因此应避免滥用浮动,下面是两种滥用浮动的典型场景。

(1) 为了使元素宽度收缩而设置浮动,由于浮动元素的宽度会自动收缩(称为主动包裹),于是就用浮动属性代替 width 属性。实际上,如果只需要改变元素的宽度,应设置 width 属性,而不是 float。

(2) 为了清除浮动的影响而浮动,清除浮动正确的做法是使用 clear 属性(例如如图 5-9 中的 Box-3),但如果对 Box-3 设置 float 属性,再设置 width 值,似乎也能达到图 5-9 中的效果,但实际上这种错误的做法将导致更多的元素受到浮动的影响而布局混乱。因此,如果要清除浮动应对应的元素设置 clear 属性,而不是 float。

5.2 浮动的应用举例

利用单个盒子浮动,可制作出图文混排及首字下沉等效果。利用多个盒子浮动,则可制作出水平导航条等效果。

5.2.1 图文混排及首字下沉效果

如果将一个盒子浮动,另一个盒子不浮动,那么浮动的盒子将被未浮动盒子的内容所环绕。如果这个浮动的盒子是图像元素,而未浮动的盒子是一段文本,那么就实现了图文混排效果。示例代码(5-8. html)如下,效果如图 5-15 所示。

```
<style>
img{
    border:1px gray dashed; margin:10px 10px 10px 0;
    padding:5px; float:left;           /* 设置图像元素浮动 */
}
p{ margin:0;
    font:14px/1.5 "宋体"; text-indent: 2em; }
</style>

<p>在遥远古希腊的大草原中,……这就是"人马族"。</p>
<p>人马族里唯独的一个例外--奇伦……</p>
```



图 5-15 图文混排效果

在图文混排的基础上让第一个汉字也浮动,同时变大,则出现了首字下沉的效果,添加的 CSS 代码(5-9. html)如下,效果如图 5-16 所示。

```
p:first-letter{font-size:3em; float:left;}
```

如果将第一个段落浮动,则出现了文章导读框效果,代码(5-10. html)如下,效果如图 5-17 所示。



图 5-16 首字下沉和图文混排效果

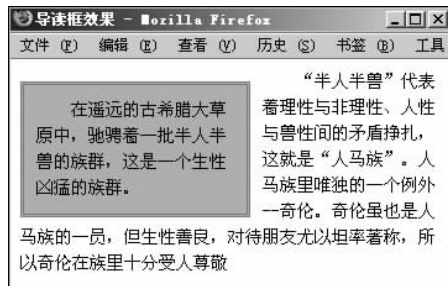


图 5-17 导读框效果

```
p{ margin:0; font-size:14px; line-height:1.5;
    text-indent: 2em; }
p:first-child{ width:160px; float:left; /* 第一个段落浮动 */
    margin:10px 10px 0 0; padding:10px;
    border:3px gray double; background:#9BD; }
```

<p class="p1">在遥远的古希腊大草原中……</p>

<p>"半人半兽"代表着理性与非理性……</p>

从以上 3 个例子可以看出,网页中无论是图像还是文本,对于任何元素,在排版时都应视为一个盒子,而不必在乎元素的内容是什么。

5.2.2 水平导航条

在 4.6.3 节中,利用元素的盒子模型制作了一个竖直导航条。如果要把这个竖直导航条变为水平导航条,有以下两种方法:

方法一,设置所有 a 元素浮动,这是因为多个元素浮动,这些元素就会水平排列。当然水平导航条一般不需设置宽度,可以把 width 属性去掉。图 5-18 是水平导航条的效果,它的结构代码(5-11.html)如下:

```
<div id="nav">
  <a href="#">首 页</a><a href="#">中心简介</a>
  ……<a href="#">技术支持和服务</a>
</div>
```



图 5-18 水平导航条

CSS 样式主要是给元素设置盒子和背景属性,代码如下:

```
#nav{font-size: 14px;}
```

```
#nav a {
    color: red; background-color: #9CF;
    text-align: center; text-decoration: none;
    padding: 6px 10px 4px; border: 1px solid #39F;
    float: left;                /* 使 a 元素浮动,实现水平排列 */
}
#nav a+a {margin-left: 5px;}    /* 设置第 2 到最后一个 a 元素有 5px 的左间距 */
#nav a: hover {color: white; background-color: #930; }
                                /* 设置前景色和背景色 */
```

提示:

(1) 本例使用了相邻选择器(#nav a+a)选中了除第 1 个 a 元素之外的其他 a 元素,再给它们设置左边距,这样就实现了第一个 a 元素没有左边距。

(2) 设置了浮动后元素将自动以块级元素显示,因此就不需要再对 #nav a 选择器添加 display: block 属性了,因为这是多余的。

方法二,将所有 a 元素的 display 属性设置为 inline-block,inline-block 元素兼有块级元素和行内元素的特点,表现在它可以像块级元素一样设置宽度和高度,同时它又像行内元素一样是从左到右排列,宽度不会自动伸展。其 CSS 关键代码如下:

```
#nav a {          ...
    display: inline-block; }          /* 将 5-11.html 的 float:left;改成这条 */
```

5.2.3 新闻栏目框

在网站的首页中,文字内容一般被组织成栏目框的形式。网站是按栏目组织内容的,因此栏目框是最常见的网页界面元素,掌握栏目框的制作是网页制作中一项重要的基本功。如图 5-19 所示的是一种简洁风格的栏目框。

基层动态	更多>>
计算机科学与技术系成功举办毕业生欢送会	2017-06-15
后勤管理处成功举办创建党员示范宿舍动员大会	2017-06-15
会计系成功举办2017届毕业生毕业典礼	2017-06-14
土木工程学院学生喜获佳绩	2017-06-14
会计系举办2017年专接本、考研、国考经验交流会	2017-06-14

图 5-19 栏目框示例

栏目框可分为栏目标题栏和内容列表区。对于栏目标题栏,常见的结构代码如下:

```
<div class="title">
    <h2>基层动态</h2>
    <span class="more"><a href="more.htm">更多 &gt;&gt;</a></span>
</div>
```

可见,栏目标题栏由两部分组成,即左边的栏目标题和右边的“更多”链接,因此需要两个 HTML 元素来存放。为了将栏目标题栏组合成一个整体,使用了一个 div 元素将这两个元素包裹起来。

对于“更多”链接,之所以将其放入一个 span 元素中,是为了将 CSS 布局样式和 CSS 文本样式分离。即对 span 元素设置布局样式,而对 a 元素设置文本样式。span 元素在这里起到了布局元素的作用。

对于内容列表区域,从语义上看,它是一个典型的无序列表,因此使用 ul 元素来描述列表区域,其结构代码如下:

```
<ul class="list_one">
  <li><a href="1.htm">计算机科学与技术系……</a><b>07-25</b></li>
  ...
  <li><a href="5.htm">会计系举办 2017 年……</a><b>07-06</b></li>
</ul>
```

然后将标题栏代码和内容列表区域代码用一个 div 元素包含起来,即得到栏目框的完整结构代码(5-12.html)如下:

```
<div class="news">
  <div class="title">
    <h2>基层动态</h2>
    <span class="more"><a href="more.htm">更多 &gt;&gt;</a></span>
  </div>
  <ul class="list_one">
    <li><a href="1.htm">计算机科学与技术系举办毕业生</a><b>07-25</b></li>
    ...
    <li><a href="5.htm">会计系举办 2017 年专接本交流会</a><b>07-06</b></li>
  </ul>
</div>
```

提示: 由于在网页中一般有多个栏目框,因此对栏目框中的元素一般设置 class 属性,而不设置 id 属性,从而使栏目框的样式代码可以被很多个风格相似的栏目框共用。

接下来设置栏目框的样式,从外表来看,栏目框的文本样式主要是设置文字大小和行距。栏目框的布局样式主要是要使栏目标题和“更多”链接分布在容器两端。新闻标题和日期也分布在容器的两端。这称为两端对齐,要实现两端对齐,主要有以下 3 种方法。

(1) 左右都浮动法: 设置左边的元素左浮动,右边的元素右浮动。这时由于两个盒子都浮动,不占据外围容器的空间,所以还必须设置外围盒子的高度,或设置外围盒子清除浮动,使它能包含住两个浮动的盒子。标题栏的 CSS 样式代码(5-12.html)如下:

```
.news{ width:420px; margin-left: 20px;}
.title {
```